I CLAIM:

1. In a computer system including an application program having a first direct driver application user interface (API), a graphics user interface (GUI) operating system having a second direct driver API, and a graphics driver application, the first and second direct driver APIs and the GUI operating system being capable of driving only a single monitor without display errors, a method of driving multiple monitors without display errors, comprising:

providing at least first and second monitors;

installing in the computer system a graphics card including at least first and second frame buffers for operating at least the respective first and second monitors;

operationally interposing between the GUI operating system and the application program a desktop manager application for generating an enhanced display area that is coextensive with at least the first and second monitors;

bypassing the graphics driver application; and

providing a direct access driver for receiving drawing instructions from the first and second direct driver APIs and transferring the drawing instructions to the first and second frame buffers, thereby enabling and accelerating multiple monitor drawing operations.

2. The method of claim 1 in which the GUI operating system includes one of a MICROSOFT WINDOWS 95 operating system, a MICROSOFT WINDOWS 98 operating system, and a MICROSOFT WINDOWS NT 4.0 operating system.

3. The method of claim 1 in which the first and second direct driver APIs include at least one MICROSOFT DIRECTX API.

4. The method of claim 3 in which the MICROSOFT DIRECTX API is not multiple-monitor aware, employs direct access to the graphics card, and is a version 3.0 or later.

5. The method of claim 1 in which any of the graphics driver application, the direct driver, and the desktop manager application are combined in a single application.

6. The method of claim 1 in which the desktop manager application converts single monitor drawing instructions to multiple monitor drawing instructions by monitoring and modifying drawing instructions that pass between the application program and the GUI operating system.

7. The method of claim 1 in which at least one of the first and second direct driver APIs issues a "GetFrameBufferAddress" call and the method further includes returning a single frame buffer address related to at least the first and second frame buffers in response to the "GetFrameBufferAddress" call.

8. The method of claim 7 further including:
tracking drawing instruction addresses;
changing a frame buffer address pointer when a drawing instruction address changes from one frame buffer to another frame buffer; and
remapping the frame buffer address pointers into the single frame buffer address.